Protein Annotation as Term Categorization in the Gene Ontology using Word Proximity Networks

Karin Verspoor, Judith Cohn, Cliff Joslyn, Sue Mniszewski, Andreas Rechtsteiner, Luis M. Rocha, Tiago Simas

Los Alamos National Laboratory, PO Box 1663, MS B256, Los Alamos, NM 87545

Email addresses:

KV: verspoor@lanl.gov
JC: jcohn@lanl.gov
CJ: joslyn@lanl.gov
SM: smm@lanl.gov
AR: andreas@lanl.gov
LMR: rocha@lanl.gov
TS: tiagos@lanl.gov

Abstract

Background

We participated in the BioCreAtIvE Task 2, which addressed the annotation of proteins into the Gene Ontology (GO) based on the text of a given document and the selection of evidence text for that annotation from the document. We approached the task utilizing several combinations of two distinct methods: one an unsupervised algorithm for expanding words associated with GO ids, and another treating annotation as categorization into the GO based on the lexical overlaps from terms in a protein's document neighborhood with terms on or associated with GO nodes.

Results

The evaluation results indicate that the method for expanding words associated with GO ids is quite powerful; we were able to successfully select appropriate evidence text for a given annotation in 38% of Task 2.1 queries by building on this method. The categorization methodology achieved a precision of 16% for annotation within the correct extended family in Task 2.2, though we show through subsequent analysis that this can be improved with a different parameter setting. Our architecture proved not to be very successful on the evidence text component of the task, in the configuration used to generate the submitted results.

Conclusion

The initial results show promise for both of the methods we explored, and we are planning to integrate the methods more closely to achieve better results overall.

Background

We participated in the BioCreAtIvE evaluation (Critical Assessment of Information Extraction in Biology). We addressed Task 2, the problem of annotation of a protein with a node in the Gene Ontology (GO, `http://www.geneontology.org`) [1] based on the text of a given document, and the selection of evidence text for that annotation. We approached the task utilizing various combinations of two distinct methods: one an unsupervised algorithm for expanding words associated with GO ids; and another treating annotation as categorization of terms derived from the sentential neighborhoods of the given protein in the given document into nodes in the GO based on the lexical overlaps with terms on GO nodes and terms identified as related to those nodes. The system also incorporates Natural Language Processing (NLP) components such as a morphological normalizer, a named entity recognizer, and a statistical term frequency analyzer. The unsupervised method for expanding words associated with GO ids is based on a probability measure that captures word proximity from co-occurrence data [2]. The categorization methodology uses our novel Gene Ontology Categorizer (GOC) methodology [3] to select GO nodes as "cluster heads" for the terms in the input set based on the structure of the GO.

BioCreAtIvE Task 2 had two subtasks for which we received evaluated results:

> **Task 2.1** – Given a (protein, document, GO id) triple, return the evidence text from the document supporting the annotation of the protein with the GO id.

Task 2.2 – Given a (protein, document) pair, return *n* annotations into the GO (GO ids) for the given protein based on the given document, along with supporting evidence text from the document for each annotation.

Methods

Pre-processing

Swiss-Prot and TrEMBL IDs were provided as input identifiers for the protein, so we needed to establish a set of names by which that protein could be referenced in the text. We made use of both the gene name and protein names that are in Swiss-Prot itself, when available, and a collection of synonyms constructed by Procter & Gamble Company. The fallback case was to use the name filled in from the EBI TrEMBL human data. A script was applied to the TrEMBL names that generated variants of strings containing mismatched punctuation and parentheticals such as "(precursor)" or "(fragment)" which were felt not to be likely to occur directly in the text. The resulting database tables were used to construct a list which was dynamically loaded from the database into a GATE [4] gazetteer processing module (which in turn compiles it into a finite state recognizer).

Additional pre-processing was performed on the document corpus. First, the original SGML documents were parsed to extract the Title, Abstract, and Body components, to normalize SGML character entities to corresponding ASCII characters (for instance, converting "′" to an apostrophe), and to remove all formatting tags apart from the paragraph markers. Subsequently, we morphologically normalized the documents using a tool called *BioMorpher*. BioMorpher is a morphological analysis tool built on the Morph tool originally developed at the University of Sheffield by Kevin Humphreys and Hamish Cunningham for general English, extended to include large exception lists for biological text as well as to handle some morphological patterns not handled by the original tool. Finally, we performed frequency analysis on the resulting terms, and selected representative terms for each document using a TFIDF filter (term frequency inverse document frequency, [5]).

Unsupervised Methodology for Expanding Words Associated with GO ids

The (protein, document, GO id) triples provided for training purposes, as well as those given as queries for Task 2.1, were used to determine sets of words related to GO ids following a methodology developed for the Active Recommendation Project at Los Alamos [6]. After document pre-processing, we divided each document into paragraphs and calculated for each document a matrix of word occurrence in the paragraphs: R: $P \times W$, where P is the set of all m paragraphs in a document, and W is the set of all n words. This is a Boolean matrix ($r_{i,j} \in \{0, 1\}$) that specifies if a given word occurred at least once in a given paragraph.

From the R matrices, we calculated a word in paragraph proximity matrix, *WPP*, for each document, using the co-occurrence probability measure below, as defined in [2]:

$$wpp(w_i, w_j) = \frac{\sum_{k=1}^{m} (r_{i,k} \wedge r_{j,k})}{\sum_{k=1}^{m} (r_{i,k} \vee r_{j,k})}$$

WPP denotes the association strength between pairs of words (w_i, w_j) , based on how often they co-occur in the paragraphs of a given document. A value of *wpp* (w_i, w_j) = 0.3, means that words w_i and w_j co-occur in the same paragraphs 30% of the time that either one of them occurs. To avoid artificially high values of *WPP*, we computed this value only if the total number of paragraphs in which either of the words occurs (the denominator of the formula) is at least 3.

We can think of *WPP* as an associative network of words. Indeed, the *WPP* matrix defines a fuzzy graph [7] where the vertices are words w_i, and the edges are probability weights *wpp* (w_i, w_j). Such a graph can also be understood as an associative knowledge structure that represents how words co-occur in a given document, and therefore as an associative model of the knowledge stored in each document in terms of its constituent words [8]. Figure 1 depicts a sub-graph of the *WPP* for one of the BioCreAtIvE documents (JBC_1999/bc005868).

[FIGURE 1 about here]

Next we set out to identify words associated with GO ids. Using the GO ids in the provided triples we retrieved the words from the GO node label. Let us refer to this set of words as W_{GO} (the red nodes in Figure 1, for GO id 0007266). For each document, we then retrieved a set of words highly associated with the words in W_{GO} in the relevant *WPP* network. Specifically, we returned the top 5 to 10 additional words with largest average value of *WPP* to all the words in W_{GO} (the green nodes in Figure 1). The additional words thus discovered were used to expand W_{GO}. Let us refer to the expanded set of words as W_{GOProx}; the additional words are not found in the respective GO node label, but co-occur highly in a given document with the words in the GO node label. This process is depicted in Figure 2.

[FIGURE 2 about here]

In Run 1 submitted for Task 2.1, which yielded a good result for this task (see below), for each (GOid, document) pair (note that we do not make use of the protein provided) ,we used its respective matrix *R* and used W_{GOProx} to recommend paragraphs as evidence text for the GO id. This was done using a vector intersection operation (step 4 in Figure 2). The columns of *R* are vectors of words occurring in a paragraph. We choose as evidence text for the GO id the paragraphs associated with the columns of *R* that yield the largest intersection with W_{GOProx}. That is, paragraphs containing the largest number of words also found in W_{GOProx} are selected.

System Operation

The architecture of the complete system is shown in Figure 3. As mentioned previously, morphological normalization, TFIDF-based term weighting, and proximity-based GO id word expansion are performed during pre-processing for each document. When executing a given query (for runs other than Task 2.1, run 1 as we will explain below), we wish to extract terms from the document neighborhood of the given protein in order to focus on terms that are most likely to be directly relevant to annotating the protein. In the runs submitted for evaluation, when we could identify references to the given protein in the document based on matching one of the known names for the protein in our database, we would extract all terms in the same sentence as each protein reference. These sets of terms together, with each term weighted by

TFIDF to represent its significance, would form the input items for subsequent processing.

[FIGURE 3 about here]

After expanding words in GO labels, we employ the term categorization method to predict GO annotations. This method is based on the GOC [3,9], which utilizes the structure of the GO to find the best nodes to "cover" or "categorize" a given set of other nodes. In GOC's original design, the set of input/query nodes was intended to be derived from the annotations of a set of gene products of interest. For BioCreAtIvE Task 2, GOC was extended first to accept weighted query items; and additionally extended to take terms as query items to be mapped to the set of GO nodes in which they appear.

A brief technical description and toy example of the base GOC's operation, and a description of its extensions for the task to handle weighted text queries, are provided in Appendix A. After identifying the set of node hits which lexically overlap with the input query, GOC traverses the structure of the GO, percolating hits upwards, and calculating scores for each GO node. GOC then returns a rank-ordered list of GO nodes representing "cluster heads" (note that we are *not* using "cluster" here in the sense of traditional clustering, e.g. *k*-means), in the end providing an assessment of which nodes best cover the input items, as well as data on which of the input terms contributed to the selection of each cluster head.

Input terms are mapped to GO nodes via one of three mechanisms:

- **Direct**: The term occurs in the node label of GO node
- **Definitional**: The term occurs in the definition text associated with GO node
- **Proximity**: The term is one of the W_{GOProx} terms related to a GO node through the proximity-based word expansion described above [2]

Direct and indirect associations are counted as distinct "hits" on a node and can be weighted differently. GOC is run on the derived query, and its output is considered an annotation of the query, which can be directly compared to the correct answers provided by the organizers (see discussion below). The GOC output is then further used to select the evidence text for the GO assignment associated with each cluster head (up to the limit of *n* annotations). To address this, we again draw on proximity measurement – in this case, the proximity of terms contributing to an annotation to individual paragraphs in the document. The closest match using the vector intersection operation (step 4, Figure 2) is selected as the evidence.

Results

We submitted 3 runs for each of tasks 2.1 and 2.2 (as well as a run for task 2.3 which was not scored). The runs consisted of the following configurations of the system:

Task 2.1

Run 1: A configuration bypassing GOC, utilizing only the GO label Word Expansion, based on proximity networks, followed by vector intersection of the columns of *R* and the expanded set of words associated with a GOid, W_{GOProx}, to discover paragraphs (essentially, the architecture of Figure 2).

Run 2: A configuration using the full system architecture including GOC, in which GOC is constrained to search for cluster heads only below the annotation given in the input query. Evidence selection consisted of a simple sentence selection algorithm, effectively an intersection of the terms in each sentence containing a relevant protein reference and the input terms (from the document neighborhood of the protein) reported by GOC to be used in the selection of the cluster head.

Run 3: Same configuration as above for annotation portion. Evidence selection used vector intersection of the columns of *R* and the input terms (from the document neighborhood of the protein) reported by GOC to be used in the selection of the cluster head. Thus, paragraphs were returned as evidence text.

Task 2.2

Run 1: A configuration using the full system architecture. Evidence selection consisting of the simple sentence selection algorithm.

Run 2: A configuration using the full system architecture. Evidence selection consisting of the paragraph selection algorithm.

Run 3: A configuration using the full system architecture, minus the sentence-based context term selection component, using instead a "fallback" scenario of selecting the top TFIDF-ranked terms in the document as the context terms. Evidence selection consisting of the paragraph selection algorithm.

Results were evaluated by professional annotators from the European Bioinformatics Institute (EBI) by considering the evidence text according to the two criteria – whether the evidence text included a reference to the correct protein, and whether the evidence text directly referenced the GO node returned as the annotation. On each of these two dimensions, the text was evaluated as "high" (correct), "generally" (generally correct, perhaps referencing the correct family of proteins rather than the protein itself, or the parent of the target GO annotation rather than the target annotation itself), or "low" (incorrect). Overall, the evidence text was judged as "perfect" if it scored "high" on both of the criteria, and as "generally" when the protein was correct but the GO reference was "generally". The GO annotations were not evaluated independently from the evidence text in the official evaluation results.

The results for the two tasks are shown in Tables 1 and 2. We were user 7, highlighted in blue. On Task 2.1, run 1, we achieved a score of either perfect or generally good for 413 of the results; this corresponds to a good result for 38% of the 1076 queries. Focusing just on perfect results, our result was 263 (24%). In this configuration, we ignored the protein altogether and focused on the GO node-paragraph relationship. Nonetheless, we received a score of "high" on the protein mention measurement for 638 of the 1050 (61%) answers we submitted. This result reflects a high coherence between GO nodes and given proteins in the given documents, at least at the level of paragraphs.

[TABLE 1 about here]

Our results for the other runs we submitted for Task 2.1 were less good, achieving a perfect or generally good score for 83/86 (runs 2/3, respectively) of the queries, or about 8%.

Our Task 2.2 results were in general not good, as shown in Table 2 (user 7, highlighted in blue). However, it was discovered after the initial evaluation results were returned that there had been a problem with the evaluation of our submissions, as well as the submissions of user 17. We were allowed to select one run for re-evaluation by the EBI annotators; we selected run 2. Table 2 shows the results after re-evaluation; approximately 5% "perfect" and 2% "generally" correct. The numbers in brackets indicate the original evaluation results for those runs. It is clear that the re-evaluation resulted in significantly more positive results, so that we can assume that the reported numbers for the runs 1 and 3 are also lower than the actual (corrected) results would indicate. We are also aware of a number of issues which contributed to our poor results, and which we have since addressed in part, and discuss below.

Discussion

There are several important general issues in the evaluation that impacted our performance.

Unknown proteins: The strategy that we follow for identifying the context window of a protein (in runs other than Task 2.1, run 1) depends on recognizing references to the protein in the text. We utilize a database of known names associated with protein IDs to pick out sentences mentioning the protein given in a query. We chose this strategy as it was straightforward to implement, and because protein reference identification was being addressed in BioCreAtIvE Task 1. The training data for Task 2 supported this strategy – a large majority (about 70%) of the training queries contained proteins that had names in our database. However, we discovered that the test data contained many protein IDs that were not yet available in SwissProt. Only 58 of the 286 (20%) proteins referenced in Task 2 evaluation queries were named in our database; 29/138 (21%) for Task 2.1 and 19/138 (14%) for Task 2.2. For the queries, only 153/1076 (14%) of Task 2.1 queries and 44/435 (10%) of Task 2.2 queries included proteins for which we had names. We were able to fall back to the names in the TrEMBL database, but these are of poor quality and usually there is only one name, not a full set of synonyms for a protein; often we did not find any occurrences of these names in the query document. This issue had a big impact on our ability to focus in on text within documents that was directly relevant to the protein of interest (see further discussion of this problem, below). On the other hand, post-hoc analysis of our (corrected) evaluation results for Task 2.2, run 2 showed that 16 of the 19 "perfect" and 8 of the 9 "generally" results actually were achieved for proteins *not* in our database. This suggests two possible problems: the names that we do have in our database are inadequate for effective protein reference identification and/or the use of a single sentence as context for terms related to the protein of interest is to narrow. We should therefore experiment with the size of left and right context windows to achieve better results.

Assessing Annotation Accuracy: The methodology followed by the evaluators of Task 2.2 focused on the evidence text selection, measuring whether the selected evidence text for a given query mentioned both the protein of interest, and the

function/process/component indicated by the target GO node. The prediction of the GO node itself was not evaluated independently of the evidence text. Our interpretation of the task was that there were two results, prediction of the GO node and selection of the evidence text. We considered the primary task of an annotator to be to correctly annotate a protein; in fact the GO and other biological knowledge repositories rarely reference anything more specific than a Pubmed ID as evidence for an annotation. Hence we considered the two results separately and focused our energy more on the GO node annotation component. While in some of the runs, our overall results were not strong, our overall performance is better when considering annotation distinctly from text selection.

Since completion of the evaluation, we have refined, improved, and measured our annotation results in a number of ways. First, there is a free parameter *s* to GOC called the *specificity*, which represents the extent to which the user values results which are "low" in the GO hierarchy, or more specific, over "high", or more abstract, results. Because GOC is itself a novel technique, at the time of the submission we had not yet refined our sense of the use of this parameter, and set it to be much higher than appropriate ($s=7$). We shall see that this was an improper choice, with stronger results for moderate levels of specificity.

Then, for each query we were instructed to provide a certain number *n* of annotations, and after the fact we were told what those correct annotations were. GOC returns a rank-ordered list usually longer then *n*, and so we cut this list off at *n* nodes, even if a correct answer might have occurred lower down in the list. Thus we end up with two sets of *n* nodes from the GO – our *n* annotation predictions and the *n* correct annotations.

To calculate our annotation accuracy, we can check how many of our answers match the correct answer exactly, but this doesn't account for "near misses", where we might return a parent, child, or sibling of the correct answer, and still wish to count this as some kind of correct response. Ultimately, this problem becomes that of measuring the amount of overlap between two sets of GO nodes, which is actually a difficult mathematical problem, which we [3,10] and others (e.g. [11]) are addressing. A detailed treatment of this subject is beyond the scope of this paper, but for our purposes, we measured "near misses" between two nodes *p* and *q* using the following categories:

- **Direct hit:** $p = q$
- **Nuclear family:** a direct hit, or *p* is a child, parent, or sibling of *q*.
- **Extended family:** a nuclear family hit, or *p* is grandparent, grandchild, cousin (grandchild of a grandparent or grandparent of a grandchild), aunt/uncle (child of a grandparent), or a niece/nephew (grandchild of a parent), of *q*.
- **Ancestor:** *p* is any ancestor of *q*.

Precision and recall as a function of specificity *s* across these different categories are shown in Figure 4. Results are especially poor for direct hits and very high specificity. A high specificity ($s = 7$) was used for all of the GOC-based runs submitted. For Task 2.2, the submitted results were therefore not as good as they might have been, with 6% precision and 5.9% recall for direct hits, 10.8% precision and 10.5% recall within the correct nuclear family, and 16.6% precision and 16.2% recall within the

correct extended family. For moderate levels of specificity at the level of nuclear and extended families, our results approach 50% precision.

[FIGURE 4 about here]

Note that due to the list cutoff issue, recall is bounded above by precision. Thus Figure 5 shows a more detailed analysis for precision only, and furthermore breaks out the family groups by their individual constituents (e.g. parents and siblings). Note that results are shown on a log scale.

[FIGURE 5 about here]

Some of the results appear impressive, for example approaching 100% for all ancestors and low specificity. This is misleading, since simply the topmost GO nodes like "biological process" and "gene ontology" are identified. However, looking at moderately "tight" neighborhoods like parents and grandparents, in family groups like nuclear and extended, reveals a moderately successful approach to automated functional annotation into the GO.

Discussion, GOC-based runs:
Due to the “unknown proteins” problem described above, the “protein neighborhood” terms input to GOC were in most instances the top TFIDF-ranked terms for the document as a whole, rather than coming from a coherent textual neighborhood around the protein. This had several implications. First, GOC may have been “overseeded” – since the input terms were derived from across the document, they may have matched very dispersed nodes in the GO. This would make it difficult for the GOC algorithm to confidently select a covering node for the input terms. Second, evidence text selection on the basis of overlap with or proximity to terms from across the document is difficult; it is unlikely that any single sentence/paragraph matches more than a few of these terms.

The overseeding may have worsened the impact of an additional difficulty. The number of terms from the GOC input set used to rank a GO node was very small – normally 1-3 terms – and only this subset of terms was passed on to the two evidence selection algorithms. The motivation underlying this approach was to enable the evidence text selection for a GO annotation to proceed on the basis of only those document terms relevant to that annotation. In practice, given the small and weakly coherent sets of terms that were generated, this created great difficulty for reliably selecting a contiguous chunk of text focused on that GO node. This would have impacted the quality of the evidence text selected, and hence our overall evaluation results. The problem could likely have been ameliorated by incorporating the strategy from Task 2.1, Run 1, utilizing all available information about the selected GO node, rather than limiting ourselves to terms from the context window.

Finally, we would like to explore the interaction between TFIDF weights and the importance of a term in the GO. Preliminary analysis suggests that there are very frequent terms in the GO with relatively high TFIDF scores in the corpus; this would unfairly value those terms in GOC and exacerbate the overseeding problem. Some adjustment of the weighting scheme to better take into consideration the terminological structure of the GO is perhaps warranted.

Discussion, Proximity network-based Word Expansion and Evidence Text selection:
While the proximity network-based word expansion proved to be a very useful technique, the evaluator comments indicated that they were often unhappy with paragraphs as the basic unit for evidence text. To address this, we envision several changes. We could apply the proximity measurements at the sentence level, rather than the word level; we could explore metrics for recognizing excessively long paragraphs and splitting them at positions of subtle topic change; or we could try to use more linguistic (structural) analysis to focus in on the core information expressed and narrow the text returned.

There are some additional ways to build on our results. We could calculate a global word proximity matrix, rather than one matrix per document, which should strengthen our confidence in the relationships between words, as well as relating any given word to more words due to consideration of its occurrence across the document corpus. We could also incorporate semi-metric analysis of the word proximities [2] to find additional (indirectly) related words, even if they do not co-occur in the corpus.

Conclusions

There is still significant room for improvement on this task, evidencing the complexities of automatic annotation of GO ids to proteins based on a single document, with complexities arising both from the structure of the GO itself and the difficulties of annotating into a large and extremely hierarchical structure, and from the ambiguous nature of text. However, the initial results show promise for both of the methods we explored, and further analysis has helped us to better understand the impact of the various parameters of the system. We are planning to integrate the two methods explored in this study more closely to achieve better results overall.

Authors' contributions

KV managed the project, developed the text pre-processing and NLP tools, built the integrated infrastructure of the complete system, and identified the GOC extensions necessary for our solution. JC provided database support, specifically for protein name management. CJ participated in the design of the solution and defined the mathematical extensions to GOC. SM was responsible for the code development of GOC and the implementation of the extensions in the code. AR worked on BioCreAtIvE Task 2.3 which was not evaluated in the end. LMR participated in the design of the proximity analysis and paragraph selection algorithms. TS developed the code for TFIDF and proximity analysis, and implemented paragraph selection. KV, CJ, and LMR contributed to the writing of the manuscript.

Acknowledgements

This work was sponsored by the Department of Energy, and by a cooperative research agreement with the Procter & Gamble Company, who also supplied us with the list of protein synonyms. We would like to thank Andy Fulmer and Jun Xu, and the LANL Protein Function Inference Group for their contributions to this work.

Appendix A: GOC and its Extensions

A synoptic overview of GOC's operation, and its extension for this task, is provided here (for full details about the base GOC, see [3]). GOC begins by casting the nodes of GO as a set *P*, which is equipped with a **partial order** (a reflexive, symmetric, anti-transitive binary relation $\leq \subseteq P^2$) as the union of all the is-a and has-part links, yielding a structure called a **partially ordered set** (or **poset**) $\mathbf{P} = (P, \leq)$. Two nodes $p, q \in P$ are called **comparable** when there is a chain in the GO between *p* and *q*, so that either *p* is a kind of *q*, or *p* is a part of *q* (denoted $p \leq q$), or *vice versa* (for $q \leq p$). Note that many paths (called **chains**) can connect two comparable nodes.

Then, features of GO nodes are cast as a set of **labels** *X*, and for example can be the gene products annotated to GO nodes, or in our case are the terms making up the phrases of each GO node. An **annotation function** $F : X \rightarrow 2^P$ then assigns to each feature (term) $x \in X$ the collection of GO nodes $F(x) \subseteq P$ with which they are associated. Altogether, we get a mathematical structure called a **POSet Ontology (POSO)** ***O*** = ***(*****P****,****X****,****F****)***.

Between all pairs of comparable nodes $p \leq q$ we define a **pseudo-distance** $\delta(p, q)$ to indicate how "high" *q* is above *p*. While many pseudo-distances are possible, we use the length of the minimum chain between them, denoted δ_m; the length of the maximal chain δ_x; the average of these $\delta_{ax} = (\delta_m + \delta_x)/2$; and the average of the lengths of all the chains between *p* and *q* denoted δ_{ap}. We also have normalized pseudo-distances $\overline{\delta}$ derived by dividing by the **height** of **P**, the size of the largest chain.

A toy example of a POSO is shown in Figure 6, where we have *P = { A, B, ..., K }, X = { a, b, ..., j },* and e.g. *F(b) = { A, E, F }*. The height is 4, and $A \leq B$ are comparable

nodes connected by three chains $A \le F \le B, A \le G \le B$, and $A \le H \le I \le B$, so that $\delta_m(A,B) = 3, \delta_x(A,B) = 4, \delta_{ax}(A,B) = 2.5, \delta_{ap}(A,B) = 2.33$, and e.g. $\bar{\delta}_m(A,B) = 3/4$.

[FIGURE 6 about here]

Given a pseudo-distance and a set of nodes of interest $Y \subseteq X$, we then want to develop a **scoring function** $S_Y(p)$ which returns the weighted rank of a node $p \in P$ based on the requested nodes *Y*. We have an unnormalized score $S_Y : P \to \mathbf{R}^+$ which returns an "absolute" number, and a normalized score $\hat{S}_Y : P \to [0,1]$ which returns a "relative" number. We also allow the user to choose the relative value placed on coverage vs. specificity by introducing a parameter $s \in \{\ldots,-1,0,1,2,3\ldots\}$, where low *s* emphasizes coverage, and high *s* emphasizes specificity. The scoring function can use either the unnormalized or normalized distance.

The four scoring functions used in the original GOC are shown in [3]. Output for the example in Fig 4 is shown in Table 3 for query *Y = { c, e, i }*, specificity values *s = -1, 1*, and *3*, doubly-normalized score $\hat{\bar{S}}$, and pseudo-distance δ_m. In addition to scoring each node, GOC identified "primary cluster heads", which are shown in bold; and "secondary cluster heads", which are cluster heads which are "above" a primary cluster head, which are labeled with *.

For the BioCreAtIvE Task 2 the following changes were made. As mentioned above, label sets *X* were allowed to be terms as well as gene products. Queries took the form of lists of terms weighted as described above, which generated a so-called **fuzzy bag** of *P* denoted $Q \subseteq P$, that is, an unordered collection of possibly duplicated nodes $p \in P$ equipped with weights $w : Q \to [0,1]$. For example, a query could be

{ (biosynthesis, 1.0), (biosynthetic catalysis, 1.0), (catalysis, 0.8),
(biosynthetic catalysis, 0.8), (protein lipidation, 0.2) }

which would map into the appropriate bag of nodes based on lexical matches between the terms and the GO node. The original scoring functions described in (Joslyn *et al.* 2004) are then modified as follows, letting $r = 2^s$:

Unnormalized Distance: $S_Q(p) = \sum_{q \in Q: q \le p} \frac{w(p)}{\delta^r(q,p) + 1}$

Normalized Distance: $\bar{S}_Q(p) = \sum_{q \in Q: q \le p} w(p)(1 - \bar{\delta}(q,p))^r$

And finally, normalized versions are derived as $\hat{S}_Q(p) = S_Q(p) / |Q|$ and $\hat{\bar{S}}_Q(p) = \bar{S}_Q(p) / |Q|$, where $|Q|$ is the size of the query, taken as the cardinality of the bag *Q*.

References

1. The Gene Ontology Cosortium: **Gene Ontology: Tool For the Unification of Biology**, *Nature Genetics* 2000, 25:1:25-29.
2. Rocha, LM: **Semi-metric Behavior in Document Networks and its Application to Recommendation Systems**. In *Soft Computing Agents: A New Perspective for Dynamic Information Systems*. Edited by V Loia. International

Series Frontiers in Artificial Intelligence and Applications. IOS Press, 2002:137-163.
3. Joslyn C, Mniszewski S, Fulmer A, Heaton G: **The Gene Ontology Categorizer**. Bioinformatics, to appear 2004.
4. Cunningham H, Maynard D, Bontcheva K, Tablan V: **GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications**. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02); Philadelphia*. 2002:168-175.
5. Witten IH, Moffat A, and Bell T: *Managing Gigabytes: Compressing and Indexing Documents and Images*. New York: Van Nostrand Reinhold; 1994.
6. Rocha LM, Bollen J: **Biologically Motivated Distributed Designs for Adaptive Knowledge Management**. In *Design Principles for the Immune System and other Distributed Autonomous Systems*. Edited by L Segel and I Cohen. Santa Fe Institute Series in the Sciences of Complexity. Oxford University Press, 2001:305-334.
7. Mordeson JN, Nair PS: *Fuzzy Graphs and Fuzzy Hypergraphs*. Springer-Verlag. 2000.
8. Rocha LM: **Automatic Conversation Driven by Uncertainty Reduction and Combination of Evidence for Recommendation Agents**. In: *Systematic Organization of Information in Fuzzy Systems*. NATO Science Series. Edited by P Melo-Pinto, HN Teodorescu and T Fukuda. IOS Press, 2003:249-265.
9. Joslyn C, Mniszewski S, Fulmer A, Heaton G: **Structural Classification in the Gene Ontology**. In *Proceedings of the Sixth Annual Bio-Ontologies Meeting (Bio-Ontologies 2003), Brisbane, Australia.* 2003.
10. Joslyn C: **Poset Ontologies and Concept Lattices as Semantic Hierarchies**, In *Proceedings of the 2004 Int. Conference on Conceptual Structures*, to appear in *Lecture Notes in Artificial Intelligence*, Springer-Verlag. 2004.
11. Deng M, Tu Z, Sun F, Chen R: **Mapping Gene Ontology to Proteins Based on Protein-Protein Interaction Data**, *Bioinformatics* 2004, 20:6:895-902.

Figures

Figure 1 - Subnetwork of *WPP* with 34 words for document JBC_1999/bc005868

The red nodes denote the words retrieved from the given GO annotation (0007266: “Rho”, “protein”, “signal”, “transduce”): W_{GO}. The blue nodes denote the words that co-occur very frequently ($wpp > 0.5$) with at least one of the red nodes: the co-occurrence neighborhood of the GO words. The green nodes denote the additional words discovered by our algorithm as described in the text. Only edges with $wpp > 0.3$ are shown.

Figure 2 - GO id Word Expansion via proximity measure

(1) For each document, a Boolean matrix of word occurrence in paragraphs (R) is created. (2) Co-occurrence proximity network *WPP* is computed. (3) Words in GO id label (W_{GO}) are expanded (W_{GOProx}) using *WPP*. (4) Intersection of vector of expanded GO node words (W_{GOProx}) with word vectors for each paragraph in the document (columns of R) : paragraph with largest intersection is returned.

Figure 3 - LANL System Architecture for BioCreAtIvE Task 2

Figure 4 - Precision vs. Recall for different values of Specificity, *s*

Paired precision (*P*) and recall (*R*) results as a function of specificity broken out by inclusive "family groups" as mentioned in the text. Note that recall is bounded above by precision, due to the need to cut off the number of GOC cluster heads considered based on the number of requested results.

Figure 5 - Precision for different values of Specificity, *s*

Log of precision as a function of *s*, broken out by the distinct (non-cumulative) family relations. For example, the precision results for "nuclear family" in Fig. 4 is the sum of direct hits from Fig. 4 and parents, children, and siblings from here.

Figure 6 - A toy example of a labeled poset

GO nodes are modeled by nodes with capital letters, with gene labels annotated to them in lower case. Note that the structure is neither a tree nor a lattice, but technically, the Hasse diagram of a poset **P**.

Tables

Table 1 - Results across all users for BioCreAtIvE Task 2.1

Evaluation results on the evidence text selected for Task 2.1. A “perfect” evaluation indicates that the evidence text refers to both the correct protein and the correct GO node. A “generally” evaluation indicates that it refers to the correct protein and that the reference to a GO node is somewhat too general. The LANL team is user 7, highlighted in aqua.

Table 2 - Results across all users for BioCreAtIvE Task 2.2

Evaluation results on the evidence text selected for Task 2.2. See legend for Table 1. In this task, evaluation of the GO node reference was done with respect to the predicted GO annotation provided by the system.

Table 3 - Original GOC output in the toy example

GOC output for values of specificity $s \in \{-1,1,3\}$. Log of precision as a function of s, broken out by the distinct (non-cumulative) family relations. For example, nuclear family in Fig. 4 is here the sum of direct hits from Fig. 4 and parents, children, and siblings from here.

Table 1:

User, Run	# results	“perfect”	“generally”
4, 1	1048	268 (25.57%)	74 (7.06%)
5, 1	1053	166 (15.76%)	77 (7.31%)
5, 2	1050	166 (15.81%)	90 (8.57%)
5, 3	1050	154 (14.67%)	86 (8.19%)
7, 1	1050	263 (25.05%)	150 (14.29%)
7, 2	1856	43 (2.32%)	40 (2.16%)
7, 3	1698	59 (3.47%)	27 (1.59%)
9, 1	251	125 (49.80%)	13 (5.18%)
9, 2	70	33 (47.14%)	5 (7.14%)
9, 3	89	41 (46.07%)	7 (7.87%)
10, 1	45	36 (80.00%)	3 (6.67%)
10, 2	59	45 (76.27%)	2 (3.39%)
10, 3	64	50 (78.12%)	4 (6.25%)
14, 1	1050	303 (28.86%)	69 (6.57%)
15, 1	524	59 (11.26%)	28 (5.34%)
15, 2	998	125 (12.53%)	69 (6.91%)
17, 1	412	0 (0.00%)	1 (0.24%)
17, 2	458	1 (0.22%)	0 (0.00%)
20, 1	1048	300 (28.63%)	57 (5.44%)
20, 2	1050	280 (26.72%)	60 (5.73%)
20, 3	1050	239 (22.76%)	59 (5.62%)

Table 2:

User, Run	# results	“perfect”	“generally”
4, 1	661	78 (11.80%)	49 (7.41%)
7, 1	153	1 (0.65%)	1 (0.65%)
7, 2	384	19 (4.95%) [1]	9 (2.34%) [1]
7, 3	263	2 (0.76%)	10 (3.80%)
9, 1	28	9 (32.14%)	3 (10.71%)
9, 2	41	14 (34.15%)	1 (2.44%)
9, 3	41	14 (34.15%)	1 (2.44%)
10, 1	120	35 (29.17%)	8 (6.67%)
10, 2	86	24 (27.91%)	6 (6.98%)
10, 3	116	37 (31.90%)	11 (9.48%)
15, 1	502	3 (0.60%)	8 (1.59%)
15, 2	485	16 (3.30%)	26 (5.36%)
17, 1	247	52 (21.05%) [1]	23 (9.31%) [0]
17, 2	55	1 (1.82%)	0 (0.00%)
17, 3	99	1 (1.01%)	1 (1.01%)
20, 1	673	20 (2.97%)	30 (4.46%)
20, 2	672	38 (5.65%)	26 (3.87%)
20, 3	673	58 (8.62%)	27 (4.01%)

Table 3:

	s = -1		*s = 1*		*s = 3*	
Rank	$\hat{\bar{S}}_Y(p)$	*p*	$\hat{\bar{S}}_Y(p)$	*p*	$\hat{\bar{S}}_Y(p)$	*p*
1	0.7672	***C***	0.5467	***H***	0.3893	***H***
2	0.6798	***1****	0.3867	***C****	0.3333	*A;**J***
3	0.6315	*H*	0.3333	*A;I;J*		
4	0.5563	*I*			0.0617	***C****
5	0.5164	*B*			0.0615	*I*
6	0.3333	*A;J*	0.2400	***B****	0.0559	*F;G;K*
7			0.2267	***1****		
8	0.2981	*F;G;K*	0.2133	*F;G;K*		
9					0.0112	*B*
10					0.0060	***1***